Noname manuscript No.
(will be inserted by the editor)

Note on the efficiency of some iterative methods for solving nonlinear equations

Miquel Grau-Sánchez · Miquel Noguera · Jose L. Diaz-Barrero

Abstract The efficiency of algorithms for solving nonlinear equations is a measure of comparison between different iterative methods. In the case of scalar equations two parameters are considered as it is well-known, but frequently in recent literature inaccurate generalizations combining these parameters are used when solving systems of nonlinear equations. Our goal in this paper is to clarify the concept of the efficiency in the multi-dimensional case. To do it we present a detailed definition of the computational efficiency. The relation between the efficiency parameters in scalar and vectorial cases is analyzed in detail and tested in two numerical examples.

Keywords: Nonlinear equations, iterative methods, divided difference, order of convergence, efficiency index, computational efficiency.

2000 Mathematics Subject Classification: 47H99, 65H10.

This work was supported by the project MTM2011-28636-C02-01 of the Spanish Ministry of Science and Innovation.

1 Introduction

To approximate a root $x^\star \in \mathbb{R}$ of a scalar equation

$$f(x) = 0, \tag{1}$$

where $f : I \subset \mathbb{R} \longrightarrow \mathbb{R}$ is a nonlinear function and I a non-empty neighborhood of $x^\star$, iterative schemes are usually employed. For instance, given $x_0 \in I$, it may be used $x_{n+1} = \phi(x_n),\ n \geq 0$. The choice of the most appropriate algorithm for solving (1) depends on its efficiency, which

M. Grau-Sánchez
Dept. of Applied Mathematics II
Technical University of Catalonia
08034 Barcelona, Spain
E-mail: miquel.grau@upc.edu

M. Noguera
Dept. of Applied Mathematics II
Technical University of Catalonia
08222 Terrassa, Spain
E-mail: miquel.noguera@upc.edu

J. L. Diaz-Barrero
Dept. of Applied Mathematics III
Technical University of Catalonia
08034 Barcelona, Spain
E-mail: jose.luis.diaz@upc.edu

links the necessary number of iterations to obtain a prefixed precision (order of convergence) to its computational cost.

Let $\mathcal{S} = \{x_n\}_{n\geq 0} \subset \mathbb{R}$ be a convergent sequence obtained using the without memory iteration function ϕ with limit $x^\star$ such that $x_n \neq x^\star$ for all $n \geq 0$. Wall [24] suggested the quantity

$$\rho = \lim_{n\to\infty} \frac{\log|e_{n+1}|}{\log|e_n|} \tag{2}$$

as the order of $\mathcal{S}$ (also the order of iteration function ϕ) provided the limit exists, where $e_n = x_n - x^*$. It is known that ρ is then equal to the R-order of $\mathcal{S}$ defined by Ortega and Rheinboldt in [15]. Another way to express the order is using the recursion

$$e_{n+1} = K e_n^\rho + O\big(e_n^{\rho+1}\big),$$

where K is the error asymptotical constant and ρ is a positive integer if ϕ is an iterative method without memory.

To compare different iterative methods, it is widely used the efficiency index suggested by Ostrowski ([16], 1960) $EI = \rho^{\,1/\alpha}$, where ρ is the local order of convergence of the method and α represents the number of the evaluations of functions necessary to carry out the method per iteration.

Other classical measure of the efficiency for iterative methods applied to scalar nonlinear equations is the computational efficiency proposed by Traub ([22], 1964) $CE = \rho^{\,1/\omega}$, where ω is the number of operations, expressed in product units, which are needed to compute each iteration. In general, if we are interested in knowing the efficiency of a scalar scheme the parameter most used is EI, instead of any combination of it with CE.

In m-dimensional case we are interested in solving systems of nonlinear equations $F(x) = 0$, where $F : D \subset \mathbb{R}^m \longrightarrow \mathbb{R}^m$ is a nonlinear function and D is a non-empty open convex domain that contains a root $x^\star \in \mathbb{R}^m$. The choice of the most suitable iterative method, $x_{n+1} = \Phi(x_n)$, depends mainly on its efficiency which also depends on the convergence order and the computational cost. We point out that the number of operations per iteration in the computational cost increases in such a way that some algorithms will not be used because they are not efficient. In this case, we say that iterative method (without memory) Φ has order of convergence $\rho \geq 1$, if

$$e_{n+1} = K e_n^\rho + O\big(e_n^{\rho+1}\big),$$

where $K \in \mathcal{L}_\rho(\mathbb{R}^m, \mathbb{R}^m)$, $e_n^\rho = (e_n, \overset{\rho}{\ldots}, e_n)^T$ and $e_n \in \mathbb{R}^m$. We remark that the order ρ previously considered is Wall's definition (2) using norms instead of absolute values and it is the R-order given in [15]. A most complete discussion about this issue can be found in [15].

It is important to define the efficiency taking into account not only the number of evaluations of scalar functions but also other aspects. When we implement an iteration of Φ it is possible to use different alternatives such as methods based on extrapolation algorithms (RRE, MPE, MMPE, ϵ–algorithms, Arnoldi, GMRES and others (see [20] and references therein)), and schemes where a system of linear equations has to be solved at each iteration. In the following the last option is chosen and the LU-decomposition plus the resolution of two linear triangular systems in the computation of the inverse operator appears. In other words, in the multi-dimensional case we have to perform a great number of operations, while in scalar case the number of operations is reduced to a very few products.

Let ℓ be the conversion factor of quotients into products (necessary time to perform a quotient in time of product units). Recall that the number of products and quotients that we need for solving a m-dimensional linear system, by using LU-decomposition is

$$\mathcal{P}_1 \;=\; \frac{m}{6}\,(2\,m^2 - 3\,m + 1) \;+\; \ell\,\frac{m}{2}\,(m-1),$$

and for solving the two triangular linear systems with ones in the main diagonal of matrix L we have $\mathcal{P}_2 = m(m-1) + \ell m$ products. Finally, the total number of products is

$$\frac{m}{6}\left(2m^2 + 3(1+\ell)m + 3\ell - 5\right).$$

To clarify concepts we consider two iterative methods of fourth-order and we study and compare the computational cost of them. The first one is a variant of the works of Ren et al. [18] and Zheng et al. [25], where the iteration function is written hereafter. Namely, for a given $x \in D \subset \mathbb{R}^m$,

$$\Phi_1 \begin{cases} y = x - \Theta^{-1} F(x), \\ X = y - \left([x, y; F] + [x + F(x), y; F] - \Theta\right)^{-1} F(y), \end{cases} \tag{3}$$

where $\Theta = [x, x + F(x); F]$ is a divided difference operator of first order. Recall that the first divided difference operator of F in $\mathbb{R}^m$ as a mapping

$$\begin{aligned} [-,-;F] \,:\, D \times D &\longrightarrow \mathcal{L}(\mathbb{R}^m, \mathbb{R}^m) \\ (x+h, x) &\longrightarrow [x+h, x\,;\, F]\,, \end{aligned}$$

which, for all $x, x+h \in D$, is defined by $[x+h\,, x\,;\, F]\,h = F(x+h) - F(x)$, where $\mathcal{L}(\mathbb{R}^m, \mathbb{R}^m)$ denotes the set of bounded linear functions (see [15, 17] and references therein).

The second iterative function is a Steffensen-like function of three-steps where the operator Θ is frozen. That is, given $x \in D \subset \mathbb{R}^m$,

$$\Phi_2 \begin{cases} y = x - \Theta^{-1} F(x), \\ z = y - \Theta^{-1} F(y), \\ X = z - \Theta^{-1} F(z). \end{cases} \tag{4}$$

Here we will consider the divided difference of first order defined in [17] by

$$[\overline{u}, \overline{v}; F]_{ij} = \frac{1}{u_j - v_j}\left(F_i(u_1, \ldots, u_j, v_{j+1}, \ldots, v_m) - F_i(u_1, \ldots, u_{j-1}, v_j, \ldots, v_m)\right), \tag{5}$$

where $\overline{u}$, $\overline{v} \in \mathbb{R}^m$. In the scalar case we have a unique definition. Namely, if u, $v \in \mathbb{R}$, then the divided difference of first order is

$$[u, v]_f = \frac{f(v) - f(u)}{v - u}.$$

2 Clarification of main concepts

In m-dimensional case we define the computational efficiency index (CEI) by (see [8, 10, 17])

$$CEI(m, \mu, \ell) = \rho^{1/\mathcal{C}(m,\mu,\ell)}. \tag{6}$$

The computational cost of the method is given by

$$\mathcal{C}(m, \mu, \ell) = \alpha(m)\,\mu + \omega(m, \ell),$$

where $\alpha(m)$ represents the number of evaluations of the scalar functions used in the evaluation of F and the first order divided operator. It is expressed in number of products because μ is a factor that converts the number of scalar evaluations into products. The function $\omega(m, \ell)$ is the number of products of the algorithm needed per iteration. Recall that parameter ℓ means that one division is equivalent to ℓ products.

In the multi-dimensional case CEI will be used to compare the efficiencies of different algorithms. Some significant discussions on the efficiency of iterative methods can also be found in [1,5,6,9,12,19].

The motivation of this note is to clarify the use of EI and CE in the vectorial case. It is not exact, and therefore inappropriate to define the computational cost by

$$\alpha + \omega,$$

and the efficiency by $E = \rho^{1/(\alpha+\omega)}$. This definition is used recursively in [2–4, 14, 23] where the above mentioned authors may not have realized that they have mixed up two entirely different issues. We modestly think that this subject should be reviewed to avoid procedures that, in our opinion, could lead to inconsistent results.

2.1 Comparison of the efficiencies of Φ_1 and Φ_2

2.1.1 The scalar case

In one-dimensional case taking $x \in I \subset \mathbb{R}$ and $\theta = [x, x + f(x)]_f$, the iteration functions Φ_1 and Φ_2 are given by

$$\phi_1 \begin{cases} y = x - \dfrac{f(x)}{\theta}, \\ X = y - \dfrac{f(y)}{[x,y]_f + [x+f(x),y]_f - \theta}, \end{cases} \quad \text{and} \quad \phi_2 \begin{cases} y = x - \dfrac{f(x)}{\theta}, \\ z = y - \dfrac{f(y)}{\theta}, \\ X = z - \dfrac{f(z)}{\theta}. \end{cases}$$

Notice that the first step of iteration functions ϕ_1 and ϕ_2 is the same and has two evaluations, $f(x)$ and $f(x + f(x))$, one product and one quotient. The second step of ϕ_1 has a new evaluation $f(y)$ and three quotients, while the second and third step of ϕ_2 has only a quotient each one since θ is computed previously.

Recall that $EI = \rho^{1/\alpha}$, and in both cases we have $\rho_1 = \rho_2 = 4$ [18, 25]. On the other hand, $\alpha_1 = 3$ and $\alpha_2 = 4$, and consequently $EI_1 = 4^{1/3} > EI_2 = 4^{1/4}$. It is interesting to observe the small number of products and quotients of these two scalar algorithms. In general, this fact is important to distinguish the difference between the scalar and vectorial case.

Another concept that changes when we work in the resolution of a system of nonlinear equations is the optimality of an iterative method. For example, following the conjecture of Kung-Traub [13], scheme ϕ_1 has optimal order since it has fourth-order and only uses three evaluation of the function f. But, this is not important in the m-dimensional case as we will see later on.

2.1.2 The m-dimensional case

Now, we study the computational cost of iteration functions Φ_1 and Φ_2 defined in (3) and (4) respectively. For the first iteration function we have $\alpha_1(m) = 3\,m^2 = 2\,m + m^2 + 2\,m\,(m-1)$ because we evaluate twice F, $F(x)$ and $F(y)$, m^2 evaluations of scalar function F in $[x, x + F(x); F]$, $m\,(m-1)$ evaluations of scalar function F in $[x, y]$ and in $[x + F(x), y; F]$. We have m^2 quotients in the computation of each divided difference operator. Scheme Φ_1 presents three operators. Moreover, we apply twice LU-decomposition and we solve two triangular linear systems twice. Hence,

$$\begin{aligned} \omega_1(m,\ell) &= 3\,m^2\,\ell + 2\,\mathcal{P}_1 + 2\,\mathcal{P}_2 \\ &= \frac{m}{3}\left(2\,m^2 + (12\,\ell + 3)\,m + 3\,\ell - 5\right). \end{aligned}$$

For iteration function Φ_2 we have three evaluations of F, $F(x), F(y)$ and $F(z)$, and m^2 evaluations of scalar functions when we compute the unique operator Θ, then we have $\alpha_2(m) = m\,(m+3) = 3\,m + m^2$. Now, it is only necessary m^2 quotients and a factorization LU because we have only one operator Θ, but we have to solve three times a double triangular linear system. That is,

$$\begin{aligned}\omega_2(m,\ell) &= m^2\,\ell + \mathcal{P}_1 + 3\,\mathcal{P}_2\\ &= \frac{m}{6}\left(2\,m^2 + (9\,\ell+15)\,m + 15\,\ell - 17\right).\end{aligned}$$

Usually, to compare the efficiency of two iterative methods it is considered the ratio between the logarithms of their corresponding CEIs. In this case, since both schemes to be analyzed have the same order, it will be suffice to study the ratio between $\mathcal{C}_2$ and $\mathcal{C}_1$. That is,

$$R_{2,1} = \frac{\alpha_2(m)\mu + \omega_2(m,\ell)}{\alpha_1(m)\mu + \omega_1(m,\ell)} = \frac{1}{2}\frac{6\mu m + 18\mu + 2m^2 + 9m\ell + 15m + 15\ell - 17}{9\mu m + 2m^2 + 12m\ell + 3m + 3\ell - 5}.$$

Equation $R_{2,1} - 1 = 0$ determines the boundary where the computational cost of one method is better than the other. Explicitly,

$$\mu(m,\ell) = \frac{1}{6}\frac{2m^2 + 3m(5\ell-3) - 9\ell + 7}{3 - 2m}.$$

For $m \geq 2$ and $\ell \geq 1$, the function $\mu(m,\ell)$ is always negative and for $\mu \geq 0$ the computational cost of Φ_1 is higher than the computational cost of Φ_2, as can be easily checked. As a consequence, Φ_2 is the most efficient.

A connected concept with CEI is the time factor (TF) defined by

$$TF = \frac{1}{\log CEI} = \frac{a\mu + \omega}{\log \rho}.$$

It is related to the elapsed time necessary to get an approximation of the root with a certain precision [10]. Moreover, if we consider the respective and analogous time factors corresponding to EI and CE, we have

$$TF = \frac{a}{\log \rho}\,\mu + \frac{\omega}{\log \rho} = TF_{EI}\;\mu + TF_{CE}.$$

3 Numerical comparison

Hereafter, the efficiency of methods previously considered, is studied for a function in both scalar and multi-dimensional case and computed correctly.

The numerical computations were performed using the MPFR library of C++ multi-precision arithmetics [7, 21] with 4096 digits of mantissa. All algorithms were compiled by `g++(4.2.1)` for i686-apple-darwin1 with `libgmp (v.5.0.2)` and `libmpfr (v.3.1.0)` libraries in a processor Intel® Xeon E5620, 2.4GHz (64-bit machine). Now, it is natural to wonder about why the computational cost of additions and subtractions does not appear in the discussion. The reason is that in this case, one product has the same computational cost as 77 additions. Furthermore, in this machine the quotient and product ratio is $\ell = 1.731$.

For each example the starting point is the same for the two methods tested. The classical stopping criterion

$$\|e_I\| = \|x_I - \alpha\| > 10^{-\nu} \quad \text{and} \quad \|e_{I+1}\| < 10^{-\nu}, \;\; \text{where} \quad \nu = 4096,$$

is replaced by

$$\|\breve{e}_I\| > 10^{-\eta} \quad \text{and} \quad \|\breve{e}_{I+1}\| < 10^{-\eta}, \;\; \text{where} \;\; \eta = [\nu(\rho-1)/\rho]\,,$$

and $\|\cdot\| = \|\cdot\|_\infty$ (see [11]). Moreover, $\breve{e}_n$ is obtained by

$$\breve{e}_n = \left(\frac{F_r(x_n)}{F_r(x_{n-1})}\right)_{1\leq r\leq m}.$$

Note that this criterion is independent of the knowledge of the root.

Table 1 Results for scalar function (7)

method	I	D_I	T_I
ϕ_1	6	3336	26.4
ϕ_2	6	2124	34.5

To evaluate the numerical efficiency of each method in the m-dimensional case we compute its factor $\tilde{\kappa}$ (see [10]). That is, for each iteration k we get the precision (number of correct decimals) D_k in computational time $\widetilde{\Theta}(D_k)$ (elapsed time from D_0 to obtain D_k correct decimals), where

$$D_k \approx -\log_{10} \|e_k\| \approx -\frac{\rho}{\rho - 1} \log_{10} \|\breve{e}_k\|.$$

We approximate the pairs $(\log D_k, \widetilde{\Theta}(D_k))$, $1 \leq k \leq I$, in the least-squares sense by a polynomial of degree one, where the slope $\tilde{\kappa}$ is computed . Namely, $\tilde{\kappa}$ is a coefficient measuring the time of execution in function of the approximate number of correct decimals, say

$$\widetilde{\Theta}(D_I) = \tilde{\kappa}(\log D_I - \log D_0).$$

Using this value of the slope we compare the time factor TF with the computed time factor $\widetilde{TF}$ defined by

$$\widetilde{TF} = \frac{\widetilde{\Theta}(D_I)}{t_p \log q} = \frac{\tilde{\kappa}}{t_p} \approx TF = \frac{1}{\log CEI},$$

where t_p is the necessary time spent by one product and $q = D_I/D_0$.

Taking into account the definition given in [24], in these experiments we calculate the computational order of convergence $\breve{\rho}$, for short PCLOC, defined in [11] by

$$\breve{\rho} = \frac{\log \|F(x_I)\|}{\log \|F(x_{I-1})\|}.$$

If $\rho = \breve{\rho} \pm \Delta\breve{\rho}$, where $\rho = 4$ is the local order of convergence and $\Delta\breve{\rho}$ is the error of PCLOC, then we get $\Delta\breve{\rho} < 10^{-3}$. This fact means that in all computations of PCLOC we obtain at least 3 significant digits and this result is a good test for the local order of convergence of the iterative methods worked in this paper.

3.0.3 Scalar example

To test the scalar case we consider the function

$$f(x) = x - e^{-x}. \tag{7}$$

We begin the computation using the guess point $x_0 = 0.1 \in \mathbb{R}$ and the solution of (7) obtained is $x^* = 0.56714\,32904\,09783\,87299\,99687\ldots$

Table 1 shows the number of iteration I for each method, the number of correct figures D_I and the elapsed time T_I measured in milliseconds. Notice that $T_I(\phi_1) < T_I(\phi_2)$ agreeing with the theoretical results.

Table 2 Results for 7-dimensional function (8)

method	I	D_I	T_I	$\tilde{\kappa}$
Φ_1	5	1768	1025.8	338
Φ_2	5	1629	494.2	164

3.0.4 Multi-dimensional example

To test the multi-dimensional case we consider the 7-dimensional function

$$F_i(\mathbf{x}) = \sum_{j=1}^{7} x_j - (x_i + e^{-x_i}) \quad \text{for} \quad 1 \leq i \leq 7, \tag{8}$$

where $\mathbf{x} = (x_1, \ldots x_7)^t$, the guess point used is $\mathbf{x}_0 = (0.1, \ldots 0.1)^t \in \mathbb{R}^7$ and the solution of (8) obtained is $x_i^* = 0.14427\,49507\,20886\,22350\,33085\ldots$ for $1 \leq i \leq 7$. The value of μ for the function F defined in (8) is the computational cost of the exponential function; that is, $\mu = 76.4$. Table 2 shows the number of iterations I for each method, the number of correct figures D_I, the elapsed time T_I measured in milliseconds and $\tilde{\kappa}$ fitted by least-square method. Note that for this function both $T_I(\Phi_2)$ and $\tilde{\kappa}(\Phi_2)$ are smaller than the corresponding values for method Φ_1 agreeing with the theoretical CEI previously studied. Furthermore, observe that Φ_2 get maximum CEI, and this fact is just the contrary to what occurred with the values of $EI(\phi_1)$ and $EI(\phi_2)$ in the scalar case.

References

1. Amat, S., Busquier, S., Grau, A., Grau-Sánchez, M.: Maximum efficiency for a family of Newton-like methods with frozen derivatives and some applications, Appl. Math. Comp. **219**, 7954–7963 (2013)
2. Babajee, D.K.R., Cordero, A., Soleymani, F., Torregrosa, J.R.: On a Novel Fourth-Order Algorithm for Solving Systems of Nonlinear Equations J. Appl. Math., Article ID 165452, 12 pages (2012)
3. Cordero, A., Hueso, J.L., Martínez, E., Torregrosa, J.R.: A modified Newton-Jarratt's composition Numer. Algor. **55**, 87–99 (2010)
4. Cordero, A., Torregrosa, J.R.: On interpolation variants of Newton's method for functions of several variables J. Comput. Appl. Math. **234**, 34–43 (2010)
5. Ezquerro, J.A., Grau, A., Grau-Sánchez, M., Hernánde,z M.A., Noguera, M.: Analysing the efficiency of some modifications of the secant method, Comp. Math. Appl. **64**, 42066–2073 (2012)
6. Ezquerro, J.A., Grau, A., Grau-Sánchez, M., Hernández, M.A.: On the efficiency of two variants of Kurchatov's method for solving nonlinear systems, Numer. Algor. **64**, 685–698 (2013)
7. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: MPFR: a multiple-precision binary floating-point library with correct rounding, ACM Trans. Math. Softw. 33, 2 (2007) Art. 13 (15 pp.)
8. Grau-Sánchez, M., Grau, A., Noguera, M.: Frozen divided difference scheme for solving systems of nonlinear equations, J. Comput. Appl. Math. **235**, 1739–1743 (2011)
9. Grau-Sánchez, M., Grau, A., Noguera, M.: On the computational efficiency index and some iterative methods for solving systems of nonlinear equations, J. Comput. Appl. Math. **236**, 1259–1266 (2011)
10. Grau-Sánchez, M., Noguera, M.: A technique to choose the most efficient method between secant method and some variants, Appl. Math. Comput. **218**, 6415–6426 (2012)
11. Grau-Sánchez, M., Grau, A., Noguera, M., Herrero, J.R.: A study on new computational local orders of convergence, Appl. Math. Lett. **25** 2023–2030 (2012)
12. Grau-Sánchez, M., Noguera, M., Amat, S.: On the approximation of derivatives using divided difference operators preserving the local convergence order of iterative methods, J. Comp. Appl. Math. **237**, 363–372 (2013)
13. Kung, H.T., Traub, J.F.: Optimal order of one-point and multipoint iteration, J. Assoc. Comp. Mach. (ACM) **21** 643–651 (1974)
14. Montazeri, H., Soleyman, F., Shateyi, S., Motsa, S.: On a New Method for Computing the Numerical Solution of Systems of Nonlinear Equations J. Appl. Math., Article ID 751975, 15 pages (2012)
15. Ortega, J.M., Rheinboldt, W.C.: Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York (1970)
16. Ostrowski, A.M.: Solutions of equations and system of equations, Academic Press, New York (1960)

17. Potra, F.A., Pták, V.: Nondiscrete induction and iterative methods, Pitman Publishing Limited, London (1984)
18. Ren, H., Wu, Q., Bi, W.: A class of two-step Steffensen type methods with fourth-order convergence, Appl. Math. Comput. **209**, 206–210 (2009)
19. Sharma, J.R., Arora, H.: On efficient weighted-Newton methods for solving systems of nonlinear equations, Appl. Math. Comput. **222**, 497–506 (2013)
20. Sidi, A.: Review of two vector extrapolation methods of polynomial type with applications to large-scale problems, J. Comput. Sci. **3**, 92–101 (2012)
21. The GNU MPFR library 3.1.0. Available in http://www.mpfr.org
22. Traub, J.F.: Iterative methods for the solution of equations, Prentice-Hall, Englewood Cliffs, New Jersey (1964)
23. Ullah, M.Z., Soleymani, F., Al-Fhaid, A.S.: Numerical solution of nonlinear systems by a general class of iterative methods with application to nonlinear Numer. Algor. (2013), DOI 10.1007/s11075-013-9784-x
24. Wall, D.D.: The order of an iteration formula, Math. Tables Aids Comput. 10 (1956) 167–168.
25. Zheng, Q., Li, J., Huang, F.: An optimal Steffensen-type family for solving nonlinear equations, Appl. Math. Comput. **217**, 9592–9597 (2011)